Министерство транспорта Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»

РУТ (МИИТ)

Институт управления и цифровых технологий

Кафедра «Цифровые технологии управления транспортными процессами»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

**ТЕМА: «Валидация»**

**по дисциплине «Корпоративные информационные системы**»

Выполнил студент группы УВА-411 И. Д. Лёвин

Принял преподаватель кафедры ЦТУТП П. О. Козьяков

Москва 2022

**Задание**

**Вариант 6**. Описать класс Поезд. С помощью стандартного API валидации с использованием стандартных аннотаций произвести валидацию класса Поезд. Валидацию провести с использованием аннотаций при описании класса и с указанием ограничений в XML.

**Текст программы на Java**

Lab4Start.java

В классе Lab4Start создаем экземпляры классов Train и PureTrain для проверки валидации. Метод класса Validation buildDefaultValidatorFactory строит и возвращает фабрику ValidatorFactory, которая, в свою очередь, используется для построения Validator методом getValidator. Далее для валидации используется метод validate который валидирует все ограничения, применяемые с объектом.

package ru.miit.lab4validate;  
  
import javax.validation.ConstraintViolation;  
import javax.validation.Validation;  
import javax.validation.Validator;  
import javax.validation.ValidatorFactory;  
import java.util.Set;  
  
  
public class Lab4Start {  
  
 public static void main(String args[]) throws Exception {  
  
 ValidatorFactory factory = Validation.*buildDefaultValidatorFactory*();  
 Validator validator = factory.getValidator();  
 try {  
 Train train = new Train();  
 train.setNomTR("222344");  
 train.setIdPoezd(0);  
 train.setType("грузовой");  
 train.setNpp(0);  
 train.setEsrStV("");  
 train.setEsrFinV("");  
 train.setVesGruz(56);  
 train.setUdl(0.1);  
 Set<ConstraintViolation<Train>> violationsTrain = validator.validate(train);  
  
 System.*out*.println("Найдены следующие нарушения: ");  
 for (ConstraintViolation<Train> viol : violationsTrain) {  
 System.*out*.println("Нарушение: " + "Поле: " + viol.getPropertyPath() + " значение: " + viol.getInvalidValue() + " " + viol.getMessage());  
 }  
  
  
 PureTrain pureTrain = new PureTrain();  
 pureTrain.setNomTR("222344");  
 pureTrain.setIdPoezd(0);  
 pureTrain.setType("грузовой");  
 pureTrain.setNpp(0);  
 pureTrain.setEsrStV("");  
 pureTrain.setEsrFinV("");  
 pureTrain.setVesGruz(56);  
 pureTrain.setUdl(0.1);  
  
 Set<ConstraintViolation<PureTrain>> violationsPureTrain = validator.validate(pureTrain);  
  
 System.*out*.println("Найдены следующие нарушения: ");  
 for (ConstraintViolation<PureTrain> viol : violationsPureTrain) {  
 System.*out*.println("Нарушение: " + "Поле: " + viol.getPropertyPath() + " значение: " + viol.getInvalidValue() + " " + viol.getMessage());  
 }  
 } finally {  
 factory.close();  
 }  
 }  
}

Train.java

Реализация класса Train с использованием аннотаций для валидации.

package ru.miit.lab4validate;  
  
import org.hibernate.validator.constraints.Range;  
  
import javax.validation.constraints.Min;  
import javax.validation.constraints.NotNull;  
import javax.validation.constraints.Size;  
  
  
public class Train {  
  
 @NotNull (message="Номер для поезда должен быть задан")  
 @Size(min = 10, message="Номер для поезда должен быть минимум 10 символов")  
 private String nomTR = null; // Номер поезда  
  
 @NotNull (message="Идентификатор для поезда должен быть задан")  
 @Min(value = 1, message = "Условная длина для поезда должна быть больше 1 условнго вагона")  
 private Integer idPoezd; // Идентификатор поезда  
  
 @NotNull (message="Тип поезда должен быть задан")  
 private String type = null; // Тип поезда  
  
 @Range(min = 1, max = 170, message="Колличество вагонов не должно быть не меньше 1 и не больше 170")  
 private Integer npp; // Колличество вагонов  
  
 @Size(min = 6, message="Станция старта для поезда должна быть минимум 6 символов")  
 private String esrStV = null; // ЕСР станции старта поезда  
  
 @Size(min = 6, message="Станция назначения для поезда должна быть минимум 6 символов")  
 private String esrFinV = null; // ЕСР станции назначения вагона  
  
 private Integer vesGruz; // Вес груза в тоннах  
 @NotNull (message="Условная длина для вагона должна быть задана")  
 @Min(value = 1, message = "Условная длина для вагона должна быть больше 1 условнго вагона")  
 private Double udl; // Условная длина вагона  
  
 public Train() {  
 }

PureTrain.java

Реализация класса PureTrain, валидация которого в xml файле.

package ru.miit.lab4validate;  
  
  
public class PureTrain {  
 private String nomTR = null; // Номер поезда  
  
 private Integer idPoezd; // Идентификатор поезда  
  
 private String type = null; // Тип поезда  
  
 private Integer npp; // Колличество вагонов  
  
 private String esrStV = null; // ЕСР станции старта поезда  
  
 private String esrFinV = null; // ЕСР станции назначения поезда  
  
 private Integer vesGruz; // Вес груза в тоннах  
  
 private Double udl; // Условная длина вагона  
  
 public PureTrain() {  
 }  
  
 public String getNomTR() {  
 return nomTR;  
 }  
  
 public void setNomTR(String nomTR) {  
 this.nomTR = nomTR;  
 }  
  
 public Integer getIdPoezd() {  
 return idPoezd;  
 }  
  
 public void setIdPoezd(Integer idPoezd) {  
 this.idPoezd = idPoezd;  
 }  
  
 public String getType() {  
 return type;  
 }  
  
 public void setType(String type) {  
 this.type = type;  
 }  
  
 public Integer getNpp() {  
 return npp;  
 }  
  
 public void setNpp(Integer npp) {  
 this.npp = npp;  
 }  
  
 public String getEsrStV() {  
 return esrStV;  
 }  
  
 public void setEsrStV(String esrStV) {  
 this.esrStV = esrStV;  
 }  
  
 public String getEsrFinV() {  
 return esrFinV;  
 }  
  
 public void setEsrFinV(String esrFinV) {  
 this.esrFinV = esrFinV;  
 }  
  
 public Integer getVesGruz() {  
 return vesGruz;  
 }  
  
 public void setVesGruz(Integer vesGruz) {  
 this.vesGruz = vesGruz;  
 }  
  
 public Double getUdl() {  
 return udl;  
 }  
  
 public void setUdl(Double udl) {  
 this.udl = udl;  
 }  
}

beans.xml

<beans xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/beans\_1\_1.xsd"  
 version="1.1" bean-discovery-mode="all">  
  
</beans>

constraints.xml

Файл constraints.xml определяет данные для объявления ограничений, используемых классом PureTrain.

<?xml version="1.0" encoding="UTF-8"?>  
<constraint-mappings  
 xmlns="http://jboss.org/xml/ns/javax/validation/mapping"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://jboss.org/xml/ns/javax/validation/mapping/validation-mapping-1.1.xsd"  
 version="1.1">  
 <bean class="ru.miit.lab4validate.PureTrain" ignore-annotations="false">  
 <field name="nomTR">  
 <constraint annotation="javax.validation.constraints.NotNull">  
 <message>Номер для поезда должен быть задан</message>  
 </constraint>  
 <constraint annotation="javax.validation.constraints.Size">  
 <message>Номер для вагона должен быть минимум 10 символов</message>  
 <element name="min">10</element>  
 </constraint>  
 </field>  
 <field name="idPoezd">  
 <constraint annotation="javax.validation.constraints.NotNull">  
 <message>Идентификатор для поезда должен быть задан</message>  
 </constraint>  
 <constraint annotation="javax.validation.constraints.Min">  
 <message>Условная длина для поезда должна быть больше 1 условнго вагона</message>  
 <element name="value">1</element>  
 </constraint>  
 </field>  
 <field name="type">  
 <constraint annotation="javax.validation.constraints.NotNull">  
 <message>Тип поезда должен быть задан</message>  
 </constraint>  
 </field>  
 <field name="npp">  
 <constraint annotation="org.hibernate.validator.constraints.Range">  
 <message>Колличество вагонов не должно быть не меньше 1 и не больше 170</message>  
 <element name="min">1</element>  
 <element name="max">170</element>  
 </constraint>  
 </field>  
 <field name="esrStV">  
 <constraint annotation="javax.validation.constraints.Size">  
 <message>Станция старта для поезда должна быть минимум 6 символов</message>  
 <element name="min">6</element>  
 </constraint>  
 </field>  
 <field name="esrFinV">  
 <constraint annotation="javax.validation.constraints.Size">  
 <message>Станция назначения для вагона должна быть минимум 6 символов</message>  
 <element name="min">6</element>  
 </constraint>  
 </field>  
 <field name="udl">  
 <constraint annotation="javax.validation.constraints.NotNull">  
 <message>Условная длина для вагона должна быть задана</message>  
 </constraint>  
 <constraint annotation="javax.validation.constraints.Min">  
 <message>Условная длина для вагона должна быть больше 1 условнго вагона</message>  
 <element name="value">1</element>  
 </constraint>  
 </field>  
 </bean>  
</constraint-mappings>

validation.xml

Файл validation.xml, в котором определяется файл для отображения ограничений.

<?xml version="1.0" encoding="UTF-8"?>  
<validation-config  
 xmlns="http://jboss.org/xml/ns/javax/validation/configuration"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://jboss.org/xml/ns/javax/validation/configuration-validation-configuration-1.1.xsd"  
 version="1.1">  
 <constraint-mapping>META-INF/constraints.xml</constraint-mapping>  
</validation-config>

**Результат**

Изображение выглядит как текст

Автоматически созданное описание

**Вывод**

В ходе выполнения лабораторной работы была произведена валидация класса Станционный путь с помощью стандартного API валидации с использованием стандартных аннотаций. Валидация компонентов решает проблему дублирования кода и излишнего запутывания классов.